TLS-Version 1.3

TLS Version 1.3, Andreas Hallof, Abteilung Datenschutz und Datensicherheit
gematik Gesellschaft für Telematikanwendungen der Gesundheitskarte mbH | Friedrichstraße 136 | 10117 Berlin

Sicherheitsziele

- Authentifizierung der Endpunkte
- Authentizität & Integrität der übertragenen Daten
- Vertraulichkeit der übertragenen Daten
- Frische der Daten (keine Replay-Angriffe)

Einordnung ins OSI-Modell & Sicherheitsziele

#	OSI-Schicht	Beschreibung	Beispiel
8	Social Layer		
7	Application Layer	Anwendungsschicht	HTTP, SMTP, IMAP, SICCT, LDAP
6	Presentation Layer	Kodierung, Datenkompression, Verschlüsselung	TLS
5	Session Layer		
4	Transport Layer	Multiplexing, Fehlererkennung und -behebung	TCP
3	Network Layer	Routing	IP
2	Data link Layer	Verbindungsebene (Sicherungsschicht)	Ethernet frames
1	Physical Layer	Bitübertragungsschicht	Ethernet-Kabel

Verbreitung TLS

https://letsencrypt.org/stats/

https://scans.io/ The Internet-Wide Scan Data Repository

443-https-tls-alexa_top1mil	443	https	tls	alexa top1mil	2017-09-20 13:17:45
443-https-tls-full_ipv4	443	https	tls	full ipv4	2017-09-15 09:11:44
25-smtp-starttls-full_ipv4	25	smtp	starttls	full ipv4	2017-09-17 00:14:52

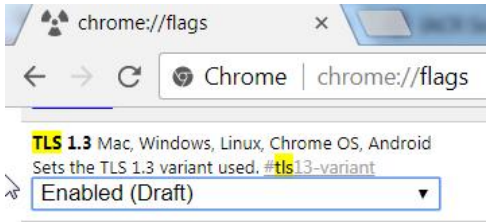
Historie

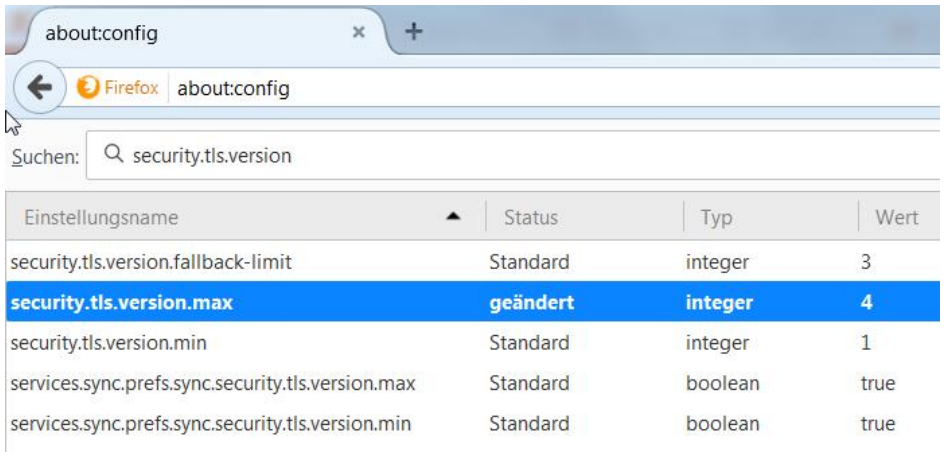
1994	SSL Version 1.0 (unveröffentlicht)
1994/95	SSL Version 2.0
1995	SSL Version 3.0 (deutliche Änderungen)
1997-1999	TLS Version 1.0
2006	TLS Version 1.1 (Sicherheitsfixes + 2003-Extensions)
2008	TLS Version 1.2 (Möglichkeit ohne SHA1/MD5 zu arbeiten, AE)
2014-	Arbeit an TLS Version 1.3 (2015 erster Entwurf)

Unterstützung des aktuellen Draft für TLS 1.3

https://en.wikipedia.org/wiki/Comparison_of_TLS_implementations#Protocol_support

Implementation	TLS 1.2[33]	TLS 1.3 (Draft)[34][35]
BearSSL	Yes	
Botan	Yes	
BoringSSL	Yes	Yes
Bouncy Castle	Yes	
CryptoComply	Yes	Yes
cryptlib	Yes	
GnuTLS	Yes	
JSSE	Yes	
LibreSSL	Yes	
MatrixSSL	Yes	
mbed TLS	Yes	
NSS	Yes[46]	Yes
OpenSSL	Yes[49]	Yes
RSA BSAFE[51]	Yes	
rustls	Yes	Yes
S2n[52]	Yes	



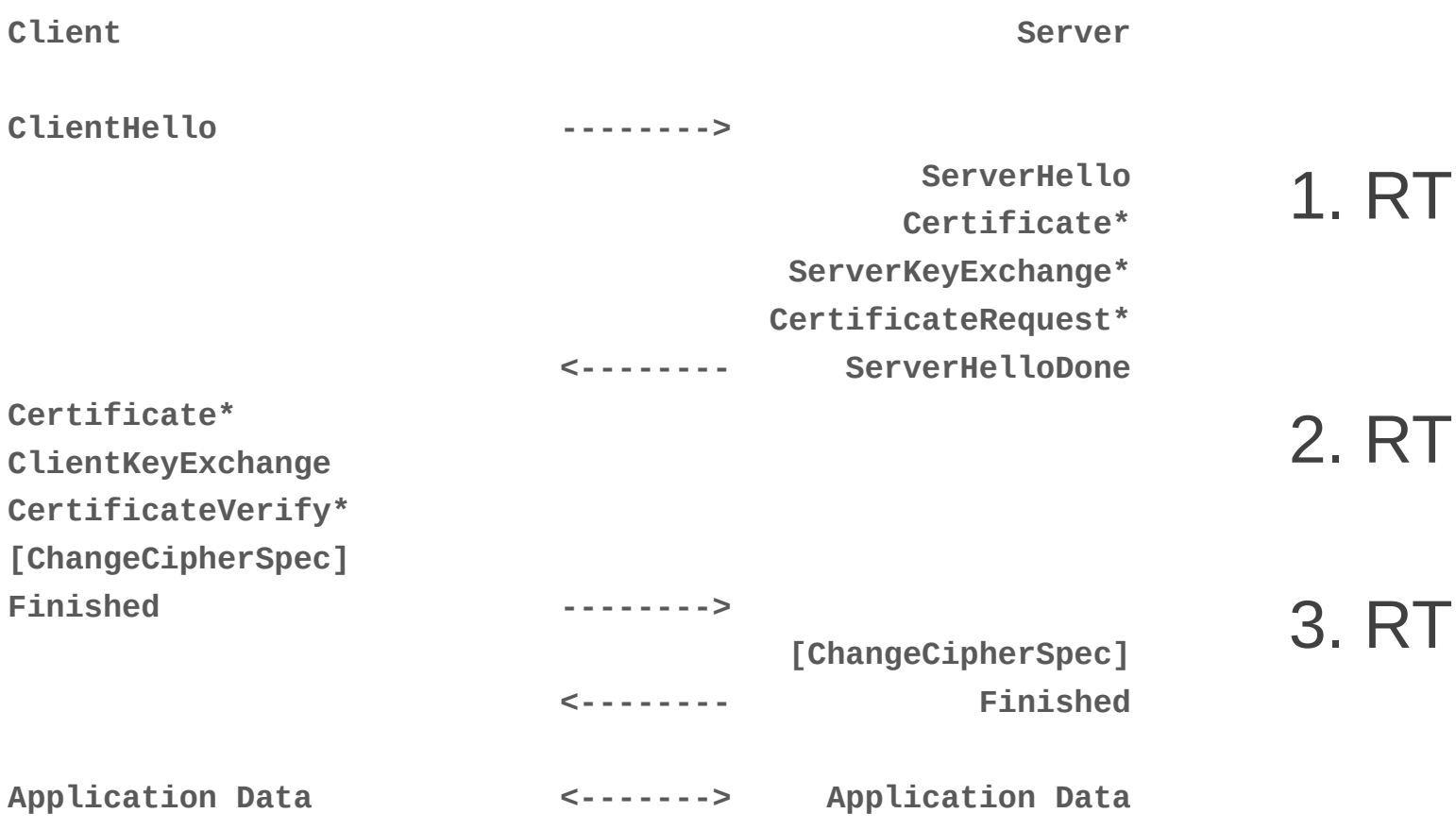


https://tlswg.github.io/tls13-spec/

https://tools.ietf.org/html/draft-ietf-tls-tls13-21

https://www.openssl.org/blog/blog/2017/05/04/tlsv1.3/

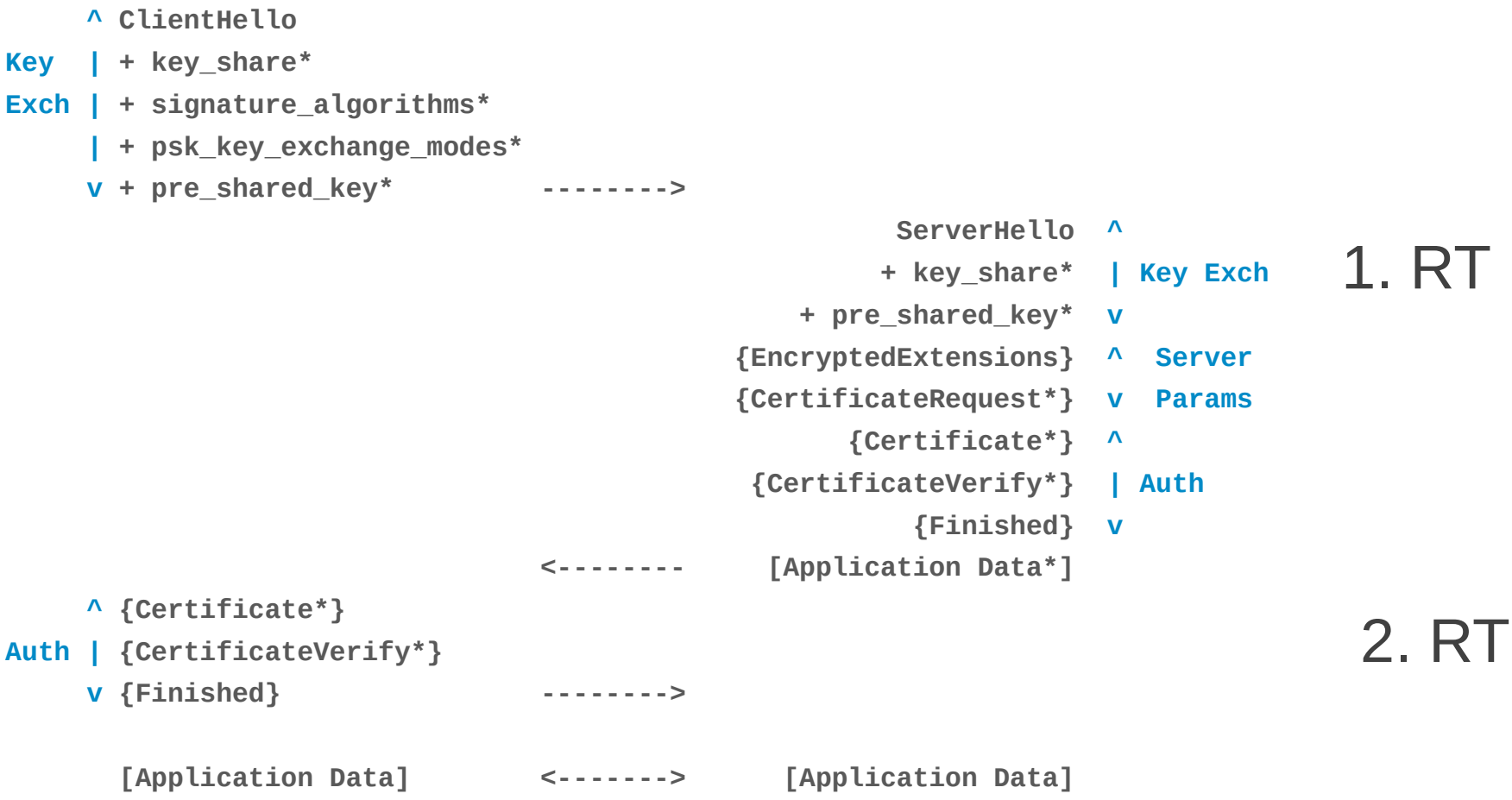
Handshake TLS 1.2 (RFC 5246)



ClientHello TLS 1.2 (RFC 5246)

```
struct {
    ProtocolVersion client_version;
    Random random;
    SessionID session_id;
    CipherSuite cipher_suites<2..2^16-2>;
    CompressionMethod compression_methods<1..2^8-1>;
    select (extensions_present) {
        case false:
            struct {};
        case true:
            Extension extensions<0..2^16-1>;
    };
} ClientHello;
```

basic full TLS handshake 1.3 (https://tools.ietf.org/html/draft-ietf-tls-tls13-21)



1. RT

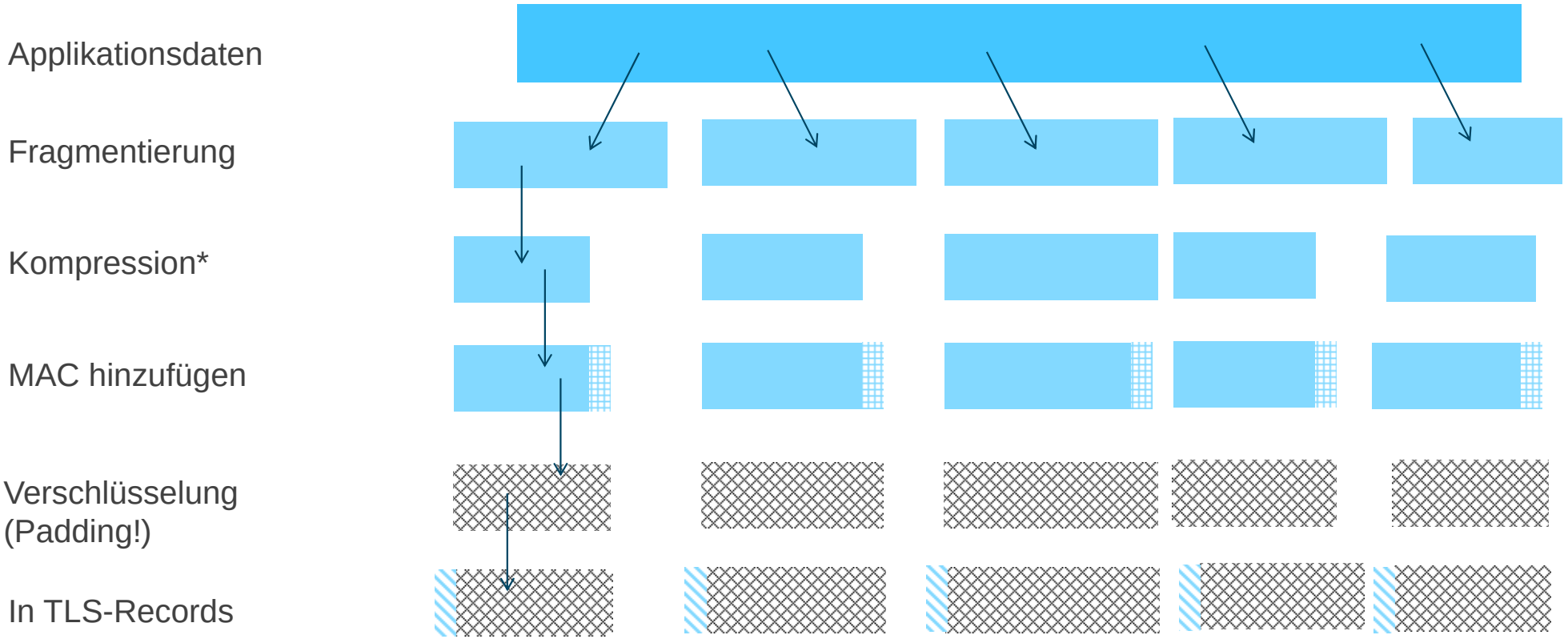
2. RT

Ziele der Überarbeitung

- Sicherheitskritische Altlasten loswerden
  (immer FS, d. h. kein Key Transport,
  nur noch AE,
  früher verschlüsseln, keine Kompression, RSASSA-PSS, Maßnahmen gegen Downgrade-Angriffe)
- Besser formal verifizierbares Protokoll
- Verbindungsaufbau mit weniger Roundtrips

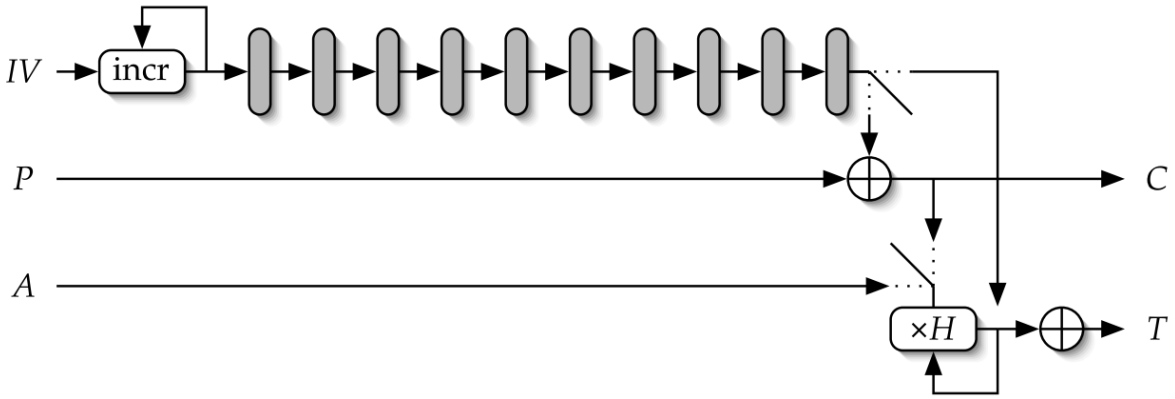
Keine Kompression / Authenticated Encryption

bei TLS 1.2



Padding PKCS#7 vs. Galois/Counter Mode

... || DD DD DD DD DD DD DD DD || DD DD DD **05 05 05 05 05** ||



pseudozufällige Nonce (IV) Erzeugung durch das TLS-Protokoll selbst

The Security and Performance of the Galois/Counter Mode of Operation (Full Version)
David A. McGrew and John Viega

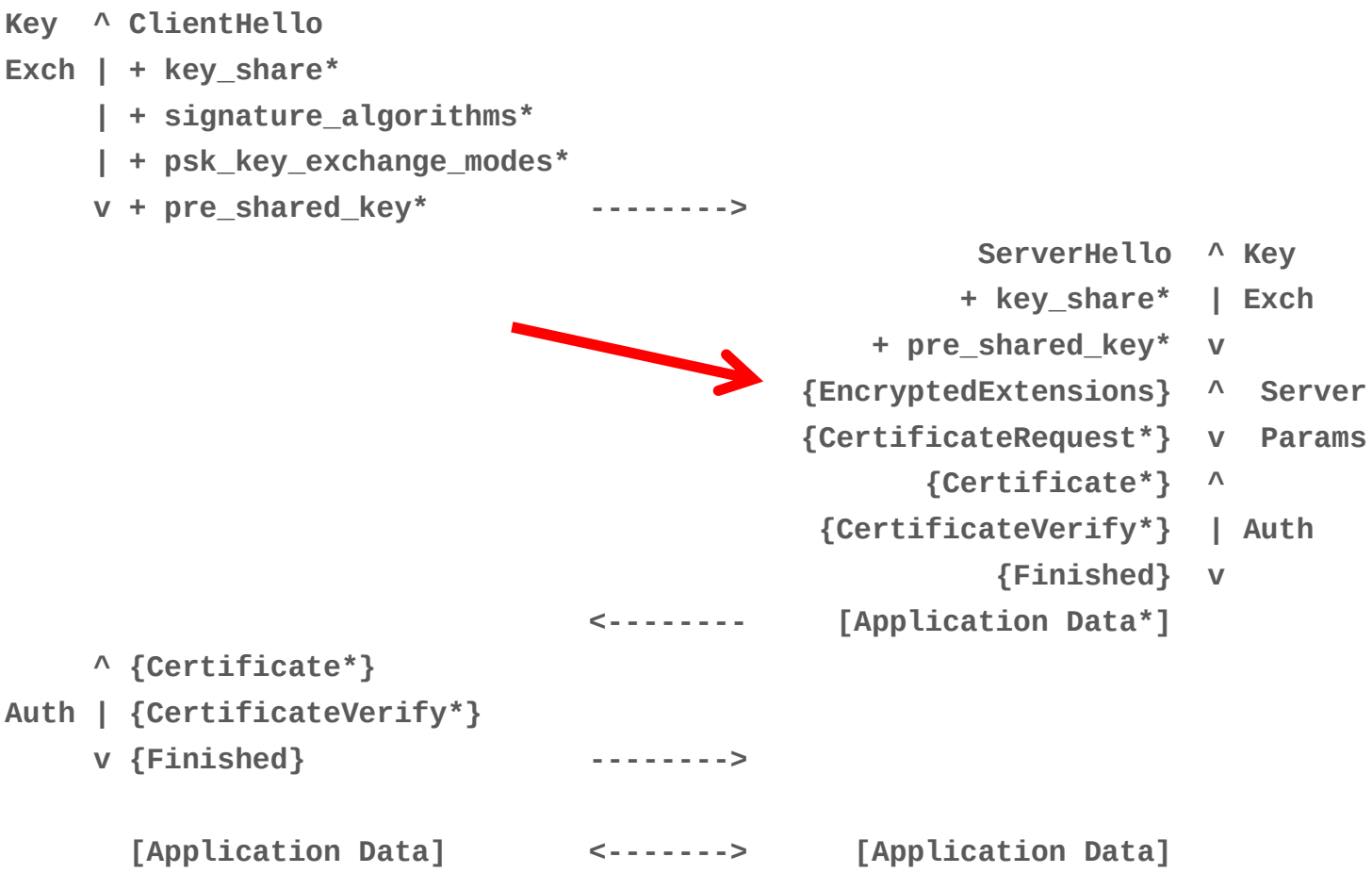
https://eprint.iacr.org/2004/193.pdf

Ciphersuiten TLS 1.3

aktuell definierte Ciphersuiten für TLS

Ciphersuite	Umzusetzen?
TLS_AES_128_GCM_SHA256	MUSS
TLS_AES_256_GCM_SHA384	SOLL
TLS_CHACHA20_POLY1305_SHA256	SOLL
TLS_AES_128_CCM_SHA256	
TLS_AES_128_CCM_8_SHA256	

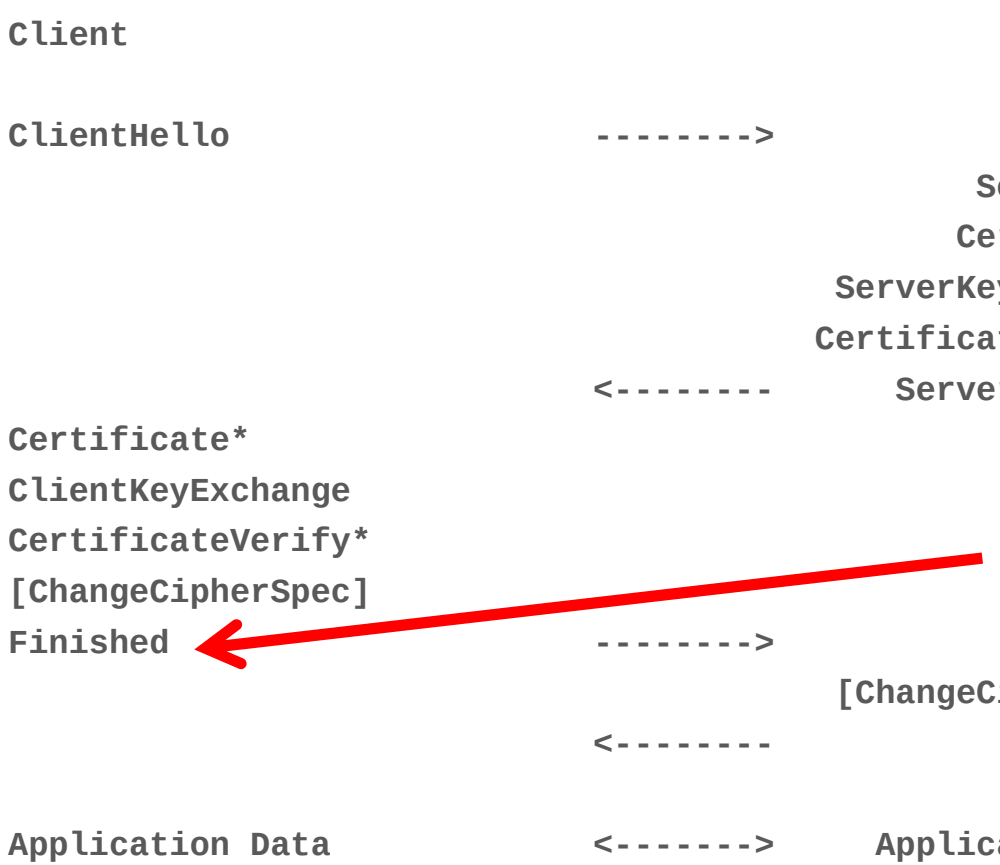
Früher Verschlüsseln (dediziertes Schlüsselmaterial)



1. RT

2. RT

Handshake TLS 1.2 (RFC 5246)

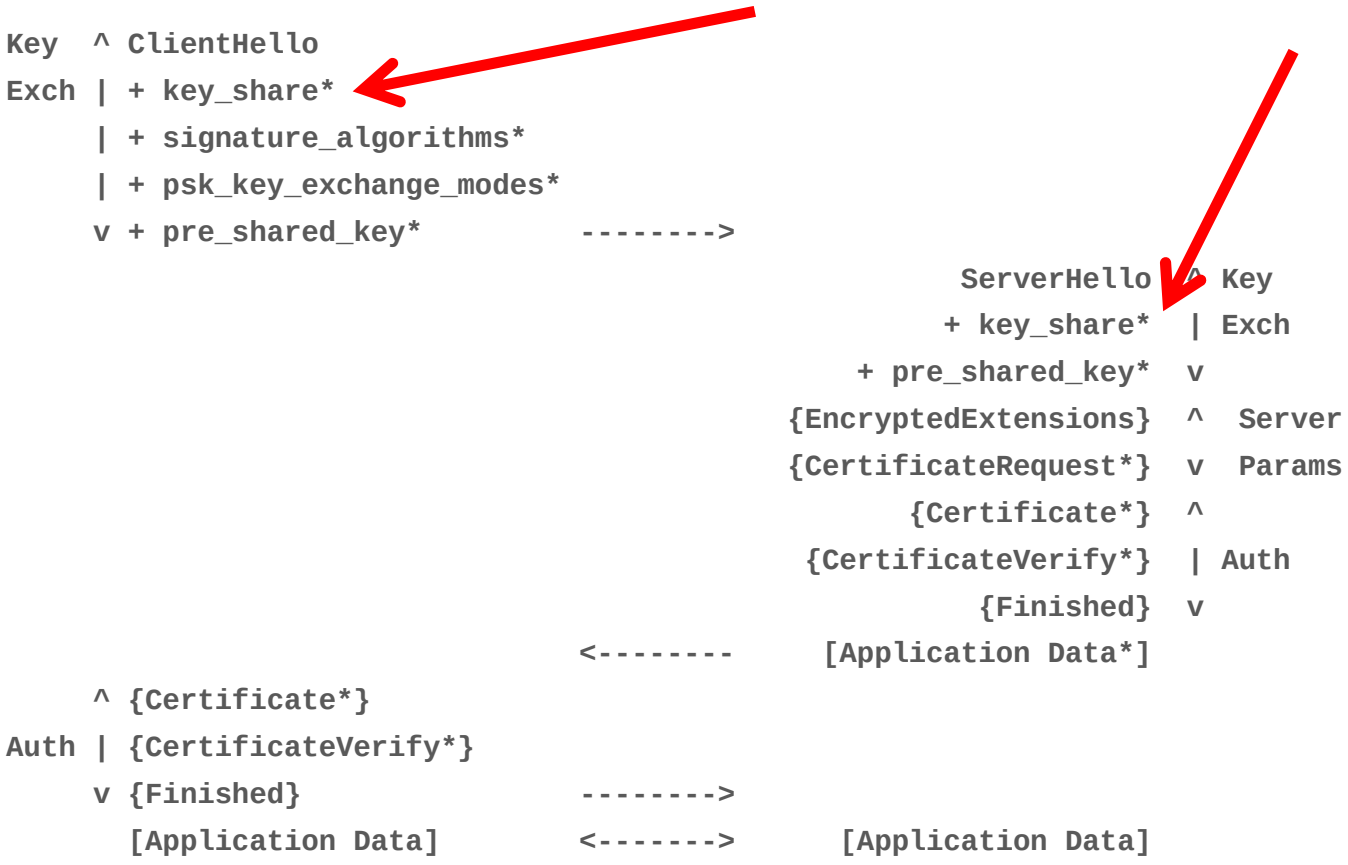


1. RT

2. RT

3. RT

Forward Secrecy



1. RT

2. RT

Forward Secrecy

Kurve P-256 für ephemeren ECDH(MUSS-Unterstützung) / Curve25519 (SOLL-Unterstützung)

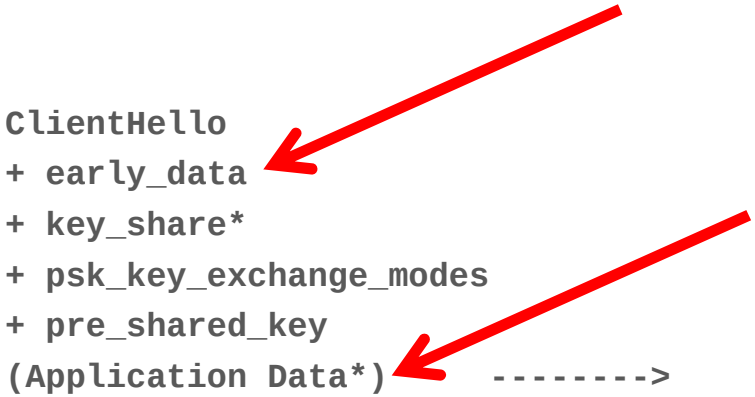
```
p = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFFFF
a = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFFFC
b = 5AC635D8 AA3A93E7 B3EBBD55 769886BC 651D06B0 CC53B0F6 3BCE3C3E 27D2604B

G =(6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0 F4A13945 D898C296,
    4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16 2BCE3357 6B315ECE CBB64068 37BF51F5)

n = FFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6FAAD A7179E84 F3b9CAC2 FC632551

y^2 = x^3 + ax + b mod p

Client: c \in [0,n-1], c*G=Pub_C

Server: s, s*G=Pub_S

c*Pub_S=s*Pub_C=c*(s*G)=s*(c*G)=Gemeinsame geheime Punkt auf P-256
```

Besser formal verifizierbar

- Deutlich mehr unterschiedliche dedizierte Schlüssel
- HKDF
- The OPTLS Protocol and TLS 1.3
  Hugo Krawczyk and Hoeteck Wee
  https://eprint.iacr.org/2015/978
- https://tools.ietf.org/html/draft-ietf-tls-tls13-21
  Abschnitt 12.2

0-RTT

Problem: Replay-Angriffe möglich



```
ClientHello
+ early_data
+ key_share*
+ psk_key_exchange_modes
+ pre_shared_key
(Application Data*)     -------->

                                        ServerHello
                                   + pre_shared_key
                                       + key_share*
                               {EncryptedExtensions}
                                       + early_data*
                                          {Finished}
                          <--------     [Application Data*]
(EndOfEarlyData)
{Finished}                -------->

[Application Data]        <------->      [Application Data]
```

ClientHello TLS Version 1.3

```
struct {
       ProtocolVersion legacy_version = 0x0303;    /* TLS v1.2 */
       Random random;
       opaque legacy_session_id<0..32>;
       CipherSuite cipher_suites<2..2^16-2>;
       opaque legacy_compression_methods<1..2^8-1>;
       Extension extensions<8..2^16-1>;
} ClientHello;
```

WIR VERNETZEN DAS GESUNDHEITSWESEN. SICHER.

TLS Version 1.3, Andreas Hallof, Abteilung Datenschutz und Datensicherheit
gematik | Gesellschaft für Telematikanwendungen der Gesundheitskarte mbH | Friedrichstraße 136 | 10117 Berlin